Question 1

(a) Consider an array of “Wooden blocks toys”. One of the features of these toys is that they are painted either Blue or Red. Devise an algorithm to keep all the Blue toys together at one end of the array and all Red toys together at the other end of the array. Is your algorithm in place? If not, what is the space complexity? What is the time complexity?

(b) Solve it for three different colors: Blue, Red and Green. . Is your algorithm in place? If not, what is the space complexity? What is the time complexity? Remember we are more concerned about the time complexity.

(c) Solve it for four different colors: Blue, Red, Green and Yellow. Is your algorithm in place? If not, what is the space complexity? What is the time complexity? Remember we are more concerned about the time complexity.

Ans:
Below algorithm works for all 3 scenarios a), b), and c) because it is using divide and conquer algorithm and it divides using colors to 2 sets in each recursive call.

```
Algorithm sortToys(A, start, stop, colorSet)
Input:
        A = Array of toys with color blue or red
        start = First pointer on left
        stop = Second pointer on right
        colorSet = set of colors to sort
Output: Sorted toys according to color

if colorSet = 1
        return

leftColorSet ← firstHalfOfColorSet
rightColorSet ← secondHalfOfColorSet
```

```
i ← start
j ← stop

while i <= j do
    while i < stop & A[i].color is in leftColorSet
        i++
    while j > start & A[i].color is in rightColorSet
        j- -
    if i < j
        swap(A, i++, j- -)

sortToys(A, start, i - 1, leftColorSet)
sortToys(A,i,stop, rightColorSet)
```

Question 2

Illustrate Quick sort. Since we do not have a random number generator, please pick a pivot so that they lead to alternating between "Good Self Call" and "Bad Self Call".

Let the number in red be the pivot number and green be the swapping numbers with either other green or red.

(a) {1, 2, 3, 4, 5, 6, 7, 8, 9}
Good Call:

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
1	**2**	**3**	**4**	**9**	**6**	**7**	**8**	**5**
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**

1	2	3	4	5	6	7	8	9
4	2	3	1	5	6	7	8	9
1	2	3	4	5	6	7	8	9
1	2	4	3	5	6	8	7	9
1	2	3	4	5	6	7	8	9

(b) {8, 7, 6, 5, 4, 3, 2, 1, 9}

8	7	6	5	4	3	2	1	9
8	7	6	5	9	3	2	1	4
1	7	6	5	9	3	2	8	4
1	2	6	5	9	3	7	8	4
1	2	3	5	9	6	7	8	4
1	2	3	9	5	6	7	8	4
1	2	3	4	5	6	7	8	9
1	2	3	4	5	6	7	8	9
3	2	1	4	5	6	7	8	9
1	2	3	4	5	6	7	8	9
1	3	2	4	5	6	8	7	9
1	2	3	4	5	6	7	8	9

1	2	3	4	6	5	7	8	9
1	2	3	4	5	6	7	8	9

(c) {9, 1, 8, 2, 7, 3, 6, 4, 5}

9	1	8	2	7	3	6	4	5
9	1	8	2	7	3	5	4	6
4	1	8	2	7	3	5	9	6
4	1	5	2	7	3	8	9	6
4	1	5	2	3	7	8	9	6
4	1	5	2	3	6	7	8	9
3	1	5	2	4	6	7	9	8
3	1	2	5	4	6	7	8	9
3	1	2	4	5	6	7	8	9
1	3	2	4	5	6	7	8	9
1	2	3	4	5	6	7	8	9

(d) {5, 1, 4, 2, 3, 9, 7, 6, 8}

5	1	4	2	3	9	7	6	8
8	1	4	2	3	9	7	6	5

3	1	4	2	8	9	7	6	5
3	1	4	2	5	9	7	6	8
3	1	4	2	5	9	8	6	7
1	3	4	2	5	6	8	9	7
1	2	3	4	5	6	7	8	9
1	2	4	3	5	6	7	9	8
1	2	3	4	5	6	7	8	9

Question 3

Let color in red be pivot value
(a) {1, 2, 3, 4, 5, 6, 7, 8, 9} k = 5
1, 2, 3, 4, $\underline{5}$, 6, 7, 8, 9
5
Since the pivot element is in 5th position and the array is sorted, 5 is the answer.

(b) {8, 7, 6, 5, 4, 3, 2, 1, 9} k = 3

8, 7, $\underline{6}$, 5, 4, 3, 2, 1, 9
5, 4, 3, $\underline{2}$, 1
5, $\underline{4}$, 3
3

(c) {9, 1, 8, 2, 7, 3, 6, 4, 5} k = 8
9, 1, 8, 2, 7, $\underline{3}$, 6, 4, 5
9, 8, 7, $\underline{6}$, 4, 5

$\underline{9}$, 8, 7
8, $\underline{7}$
8

(d) {5, 1, 4, 2, 3, 9, 7, 6, 8} k = 5
5, 1, 4, 2, $\underline{3}$, 9, 7, 6, 8
5, 4, 9, $\underline{7}$, 6, 8
5, 4, $\underline{6}$
5, $\underline{4}$
5

Question 4

Let us redefine “Good Self Call” and “Bad Self Call”

Good self-call: the sizes of L and G are each less than 2n/3 (normal division)
Bad self-call: one of L and G has size greater than or equal to 2n/3.

(a) Repeat the calculations shown in Slides 15, 16 and 17. (You need not draw pictures).
7, 2, 9, 4, 3, 7, 6, 1

7	**2**	**9**	**4**	**3**	**7**	**6**	**1**
7	**2**	**9**	**1**	**3**	**7**	**6**	**4**
3	**2**	**9**	**1**	**7**	**7**	**6**	**4**
3	**2**	**1**	**9**	**7**	**7**	**6**	**4**
3	**2**	**1**	**4**	**7**	**7**	**6**	**9**

3	4	1	2	7	7	9	6
1	4	3	2	6	7	7	9
1	2	3	4	6	7	7	9
1	2	4	3	6	7	9	7
1	2	3	4	6	7	7	9

(b) Are you able to derive the same results in Slides 16 and 17? If not, why?
Ans: The final result is the same but depending on the pivot element the steps for sorting could be different.